

ROD testing

ATLAS SCT and pixel off-detector electronics PDR

31 July, 2000

Damon Fasching

The University of Wisconsin, Madison

The ROD is a complex object:

- 12 layer 9U board
- 17000 vias
- ≈ 5000 “blind” connections (BGA)
- 5 DSP processors performing a variety of tasks
- 12 FPGAs with 40000 lines of VHDL
- various operating environments and modes
 - instantaneous 4 Gbps input data rate, including decoding
 - sustained 1.3 Gbps throughput, including event building
 - a lot of requirements, 20 page FRD

Commissioning will require a thorough, systematic test plan.

* Note, this presentation should be consumed along with M. Nagel's slides.

There are many items which may be considered part of ROD testing.

- several independent eye checks of the schematics
- VHDL simulation
- checks performed by the board manufacturer
 - test features designed onto the board
 - dedicated code for board integrity checks
- plan to build dedicated test boards to provide known data to the BOC-ROD system with a lot of flexibility for off-detector system tests (Cambridge is evaluating the possibility of taking this on. Refer to M. Goodrick's presentation.)

1. VHDL simulation

- waveform simulator largely for debugging small code blocks during initial VHDL development
- detailed simulation of entire data path done in conjunction with C model of the algorithms

- C model starts at the hit generation.
- It provides input data streams and output data at the location of each memory in the data path.
- Data from these files can be fed into the VHDL simulation at any point in the data path.
- The output from the VHDL simulation can be compared with the output of the C simulation at any downstream point.
 - ⇒ Refer to J. Joseph's presentation.
- full timing
 - again done in conjunction with C simulation data
 - takes into account propagation delays, etc
- ⇒ Refer to J. Joseph's presentation.

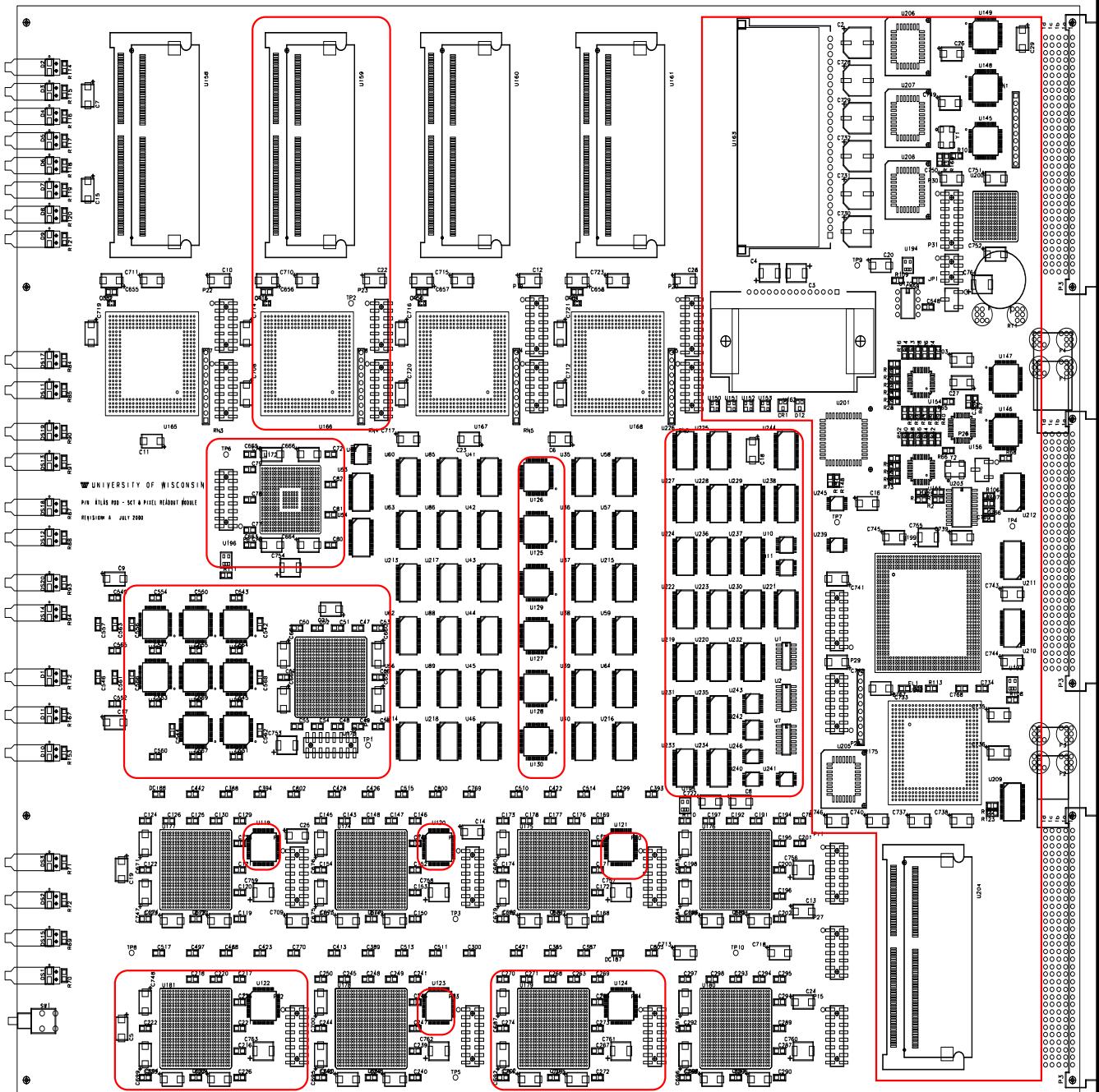
2. checks performed by the board manufacturer

- bed of nails for the prototype
- flying probe possible for production model

3. initial by-hand checks, power-up and connectivity (1st on a partially loaded board)

- Does program and reset manager FPGA (**PARM**) configure OK and hold all FPGAs/DSPs in reset?
- Bring ROD resources interface FPGA (**RRIF**) up.
 - RRIF sets up VME interface chipset.
 - Check VME communication to verify VME chipset, geographical addressing, etc.
 - Check communication with RRIF, PARM and FPGA configuration FLASH.
- **Release master DSP from reset. It will boot from its FLASH.**
 - Check communication with master DSP. Simple read and write to internal memory space checks HPI.
 - Check communication between master DSP and RRIF by host write and readback of RRIF via master DSP (HPI and EMIF).
 - Check master DSP SDRAM (connectivity, memory test).
 - “Echo” type primitive checks communication protocol, using most of the above tested data paths.

initial loading



- **Release the slave DSP from reset, download program image.**
 - Check HPI access to slave DSP memory space.
 - Burst dummy data into SDRAM and read back to check 80 MHz operation.
 - **In turn, release each FPGA from reset, configure it and check r/w access via ROD bus.**
 - **At this point, power up and boot of the ROD has been verified and access to the memory space of the partially loaded board has been checked. It is time to start adding the buffers and latches which allow access to the data path, including debug memories.**
- #### 4. **test features on the board**
- **Debug memories in the data path with r/w access from VME allow feeding test vectors through any part of the data path.**
 - These test vectors can include those generated by the C simulation which were used in the VHDL simulation phase.
 - It may be difficult to generate a set of test vectors which respects the data format and provides an exhaustive test of connectivity including flow control and other control signals. Some simple VHDL code which will be used only for testing is planned.

- All of the FPGAs and DSPs have standard JTAG signals brought out to headers.
- Buffers on data paths allow access to a many of the BGA connections.
- **dedicated code for integrity checks of ROD signal paths**
- Once the power-up/boot design and the layout have been verified, unknown problems on subsequent boards should be the (hopefully) occasional bugs due to fabrication yield. It is desirable at that point to automate board check-out as much as possible, esp. for production.
- Of course before powering on any board it should be scanned by eye for things that would make it blow up on power-up.
- host processor code to check
 - DSP access from host via HPI
 - SDRAM access from host via HPI and EMIF (for download of primitive lists)
 - RRIF-DSP communication via HPI and EMIF. This should be checked from the host. If it does not work, the DSP will not be able to communicate errors and reply data via the normal communication protocol.

- RRIF access from the host
- PARM access from the host
- test of master DSP SDRAM
- check out of VME reset function

- **master DSP code to**

- write to / read back from each ROD bus object
- test of slave DSP SDRAM
- log errors

⇒ This code would be executed upon receipt of a primitive command.

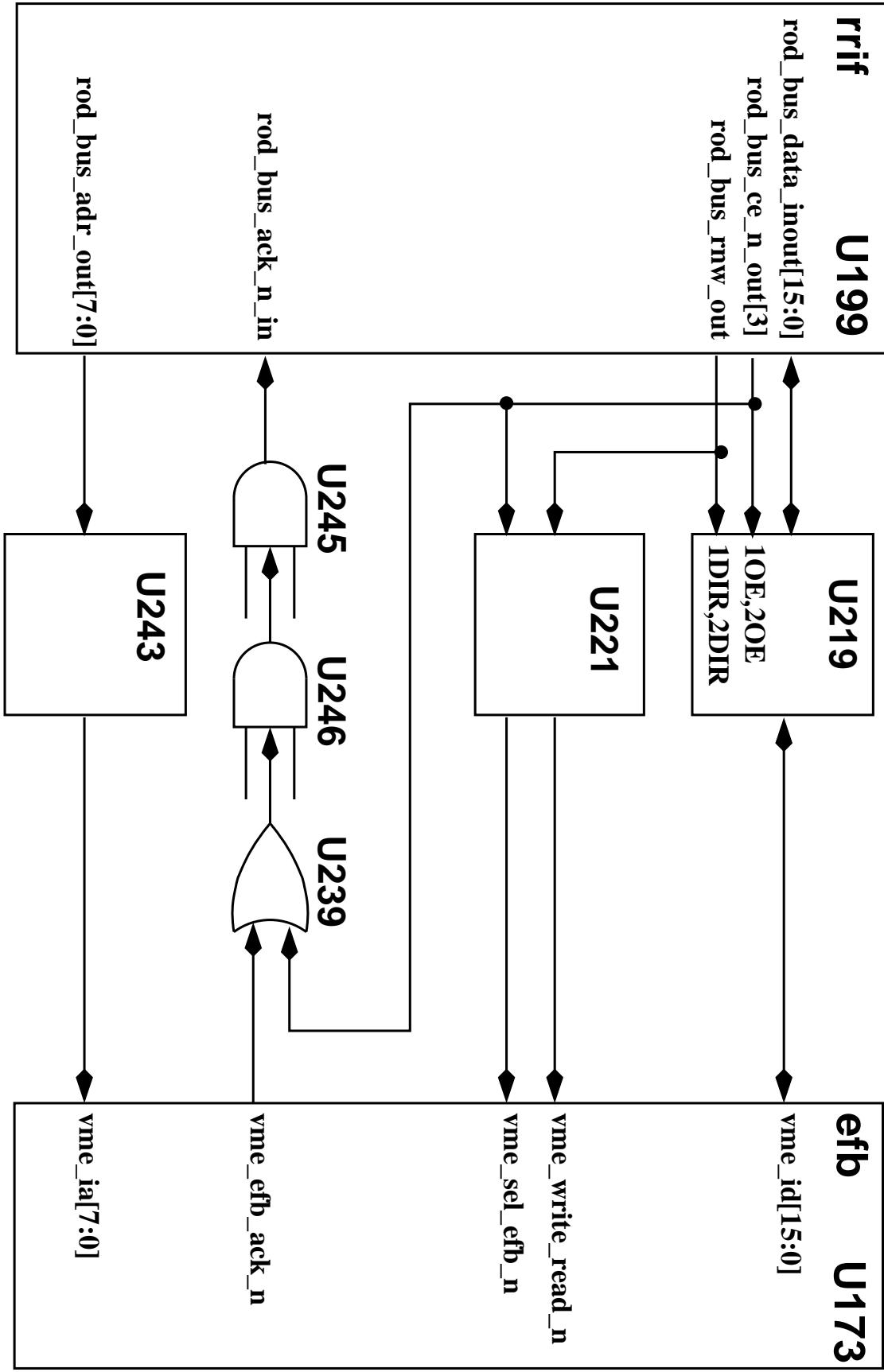
⇒ The test could be performed by the host, but it is more natural to maintain the code on the DSP, where the internal ROD memory map resides.

- **slave DSP code to**

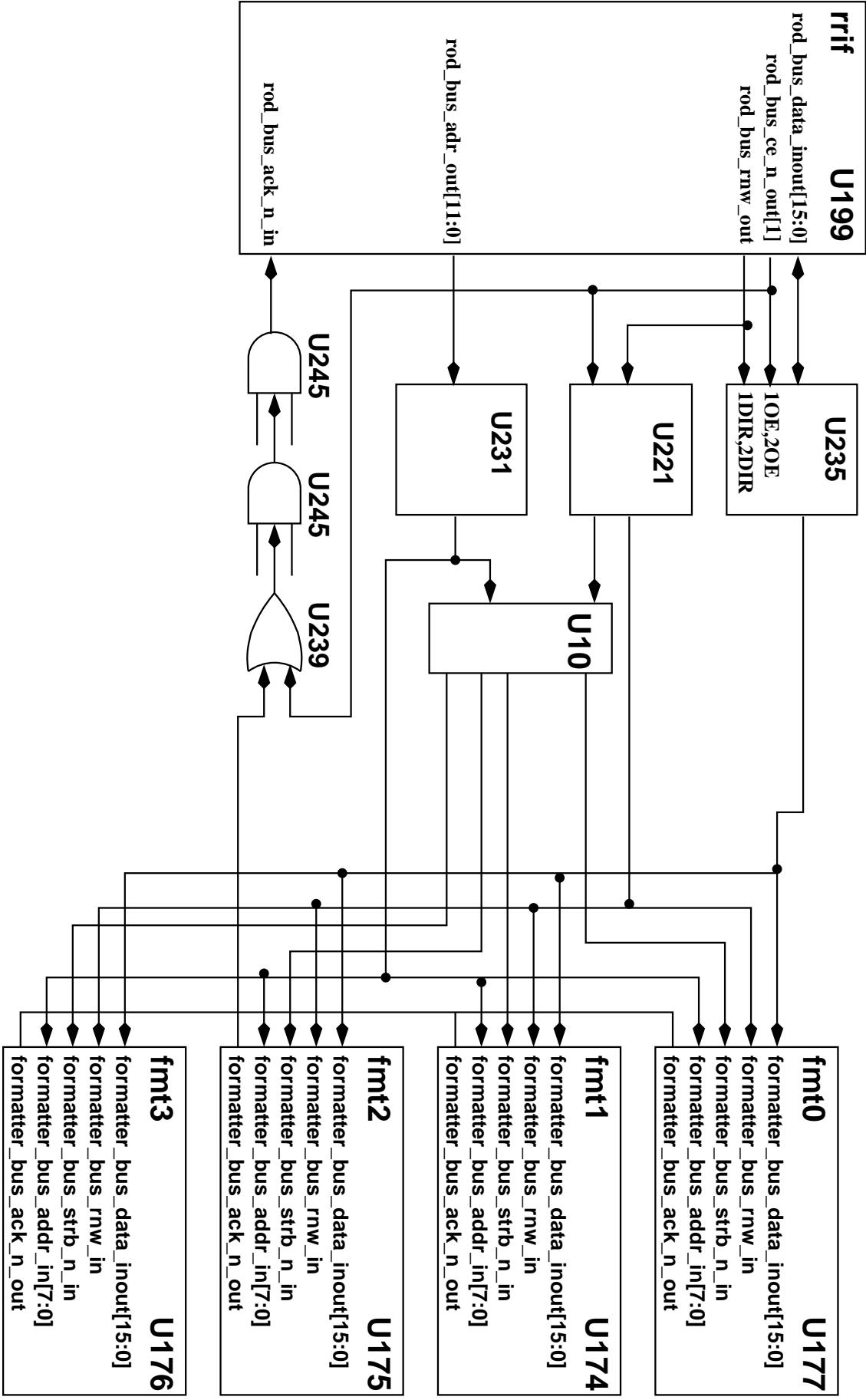
- write to / read back from the router FPGA
- burst data to and back from the SDRAM at 80 MHz
- log errors

⇒ This code would also be executed upon receipt of a primitive command.

efb Bus

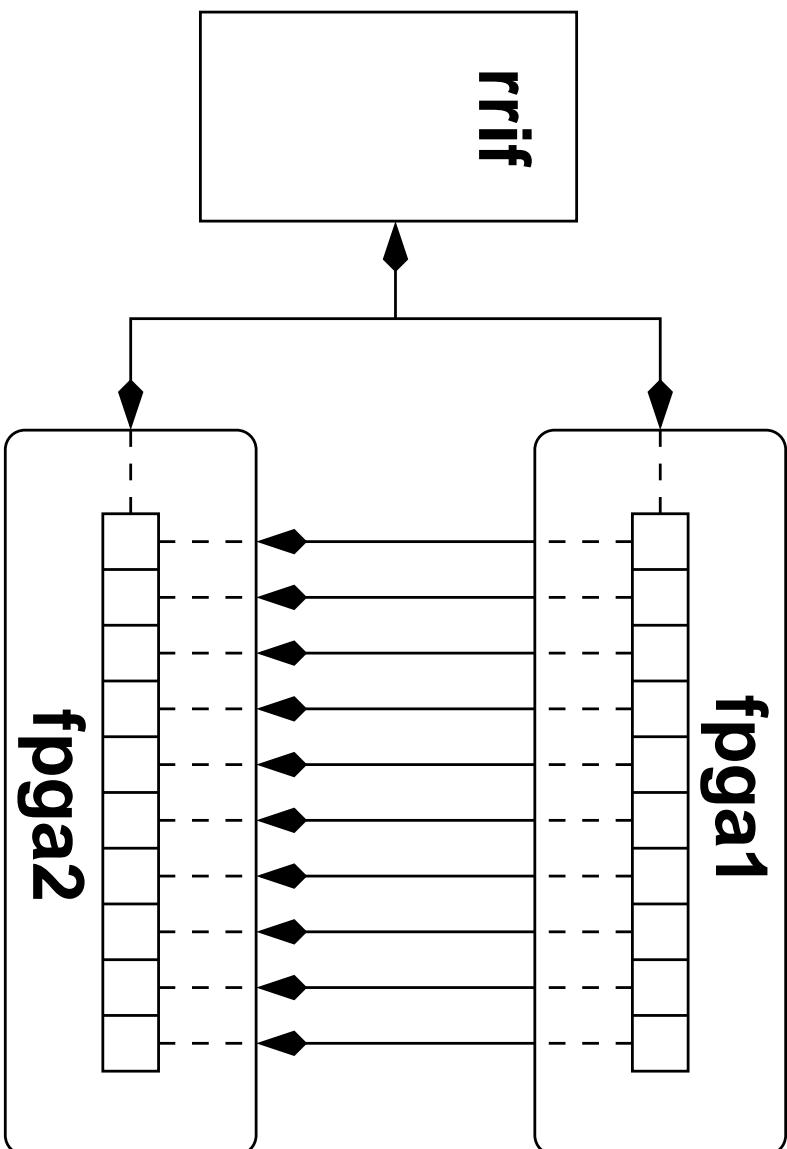


formatter A Bus

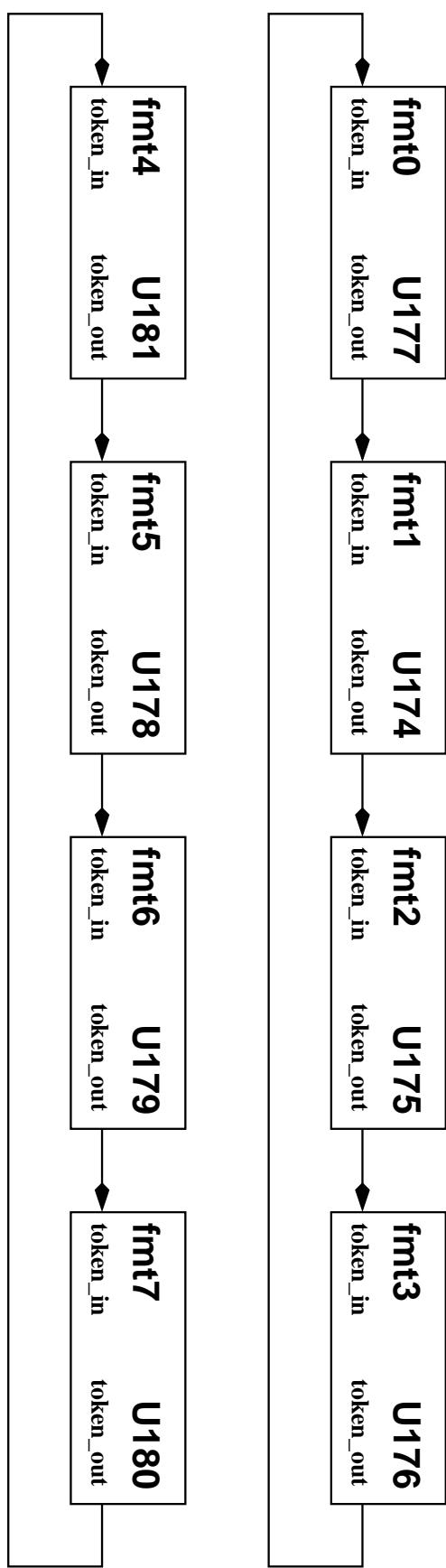


- **dedicated VHDL to check FPGA to FPGA connectivity**

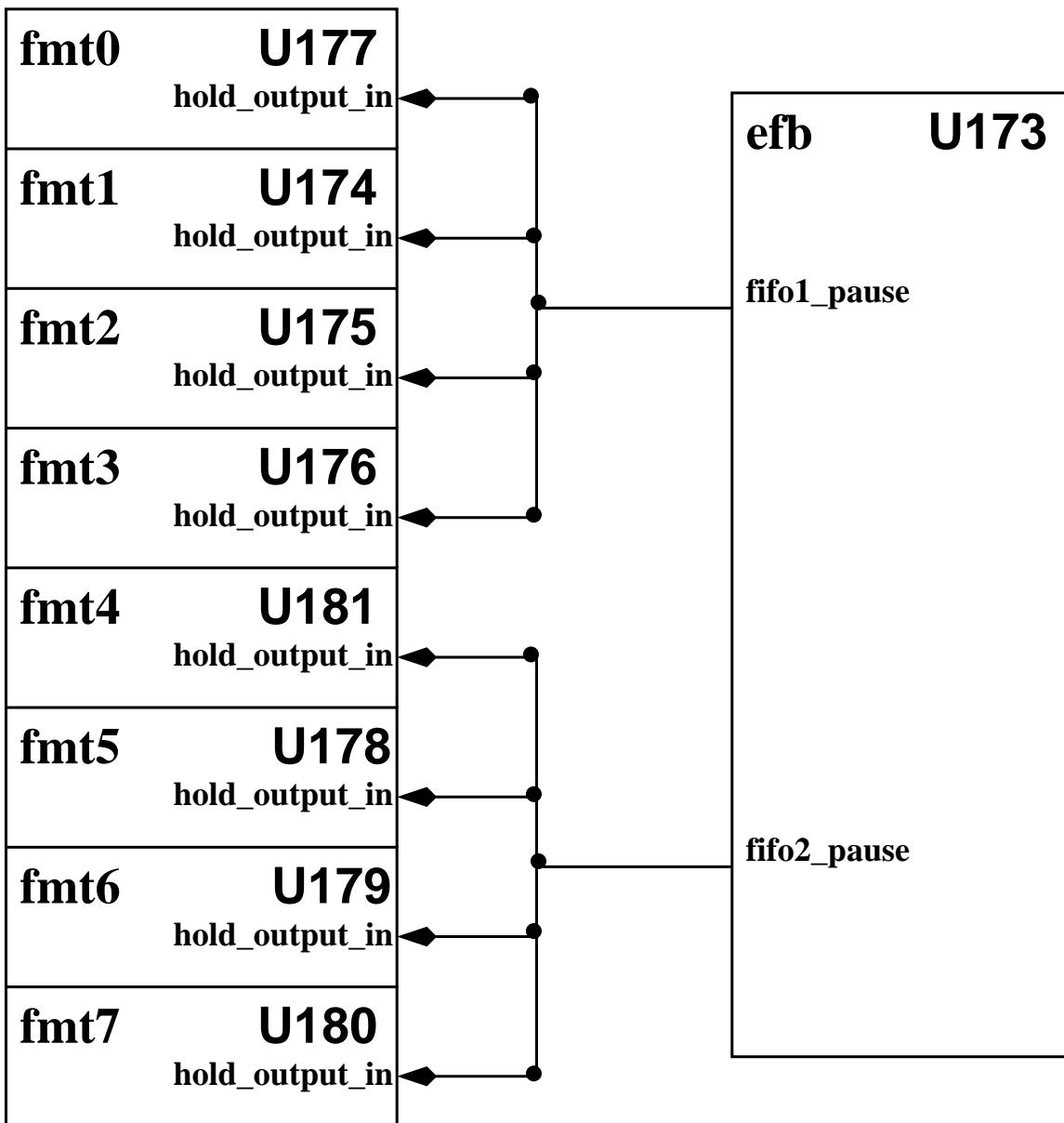
A simple scheme would be to define a read/write register in each FPGA which is accessed from the rod bus. A value written to one FPGA would be asserted on the I/O pins connected to the other. The machinery to read and write internal FPGA registers exists in the standard VHDL. Mapping an internal register to I/O pins does not require much effort. This is a static test but could be modified to be dynamic.

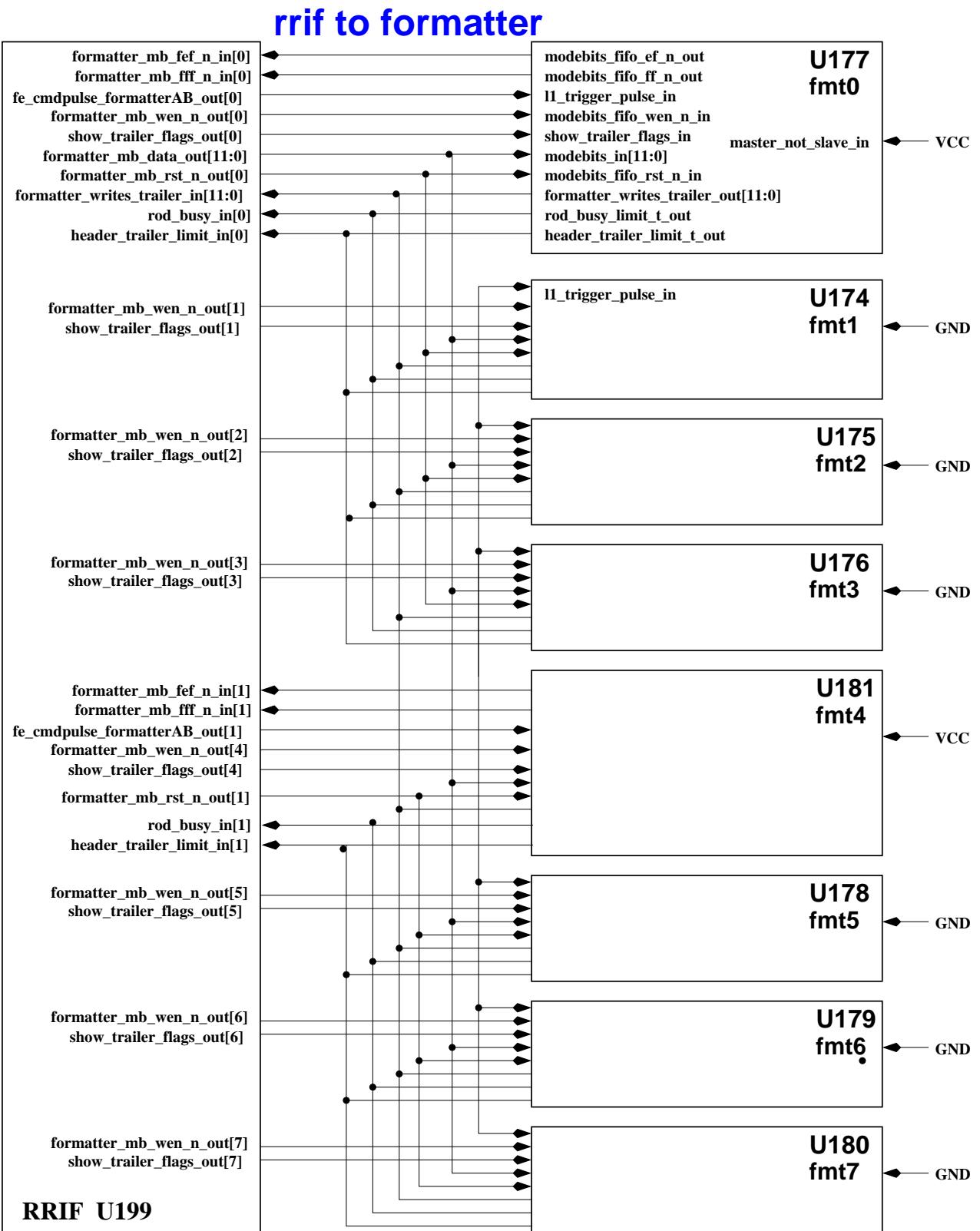


formatter-formatter



formatter-efb



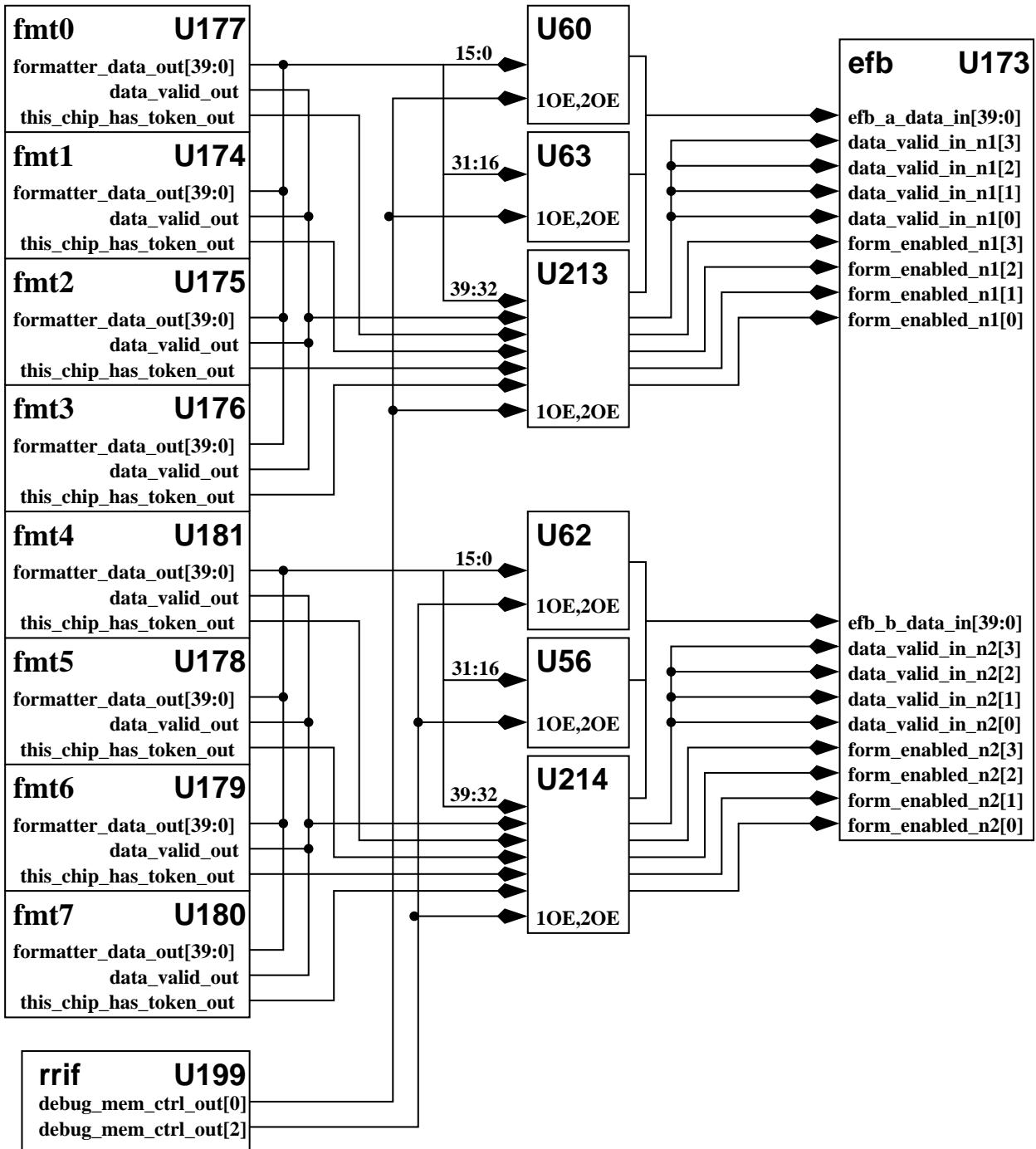


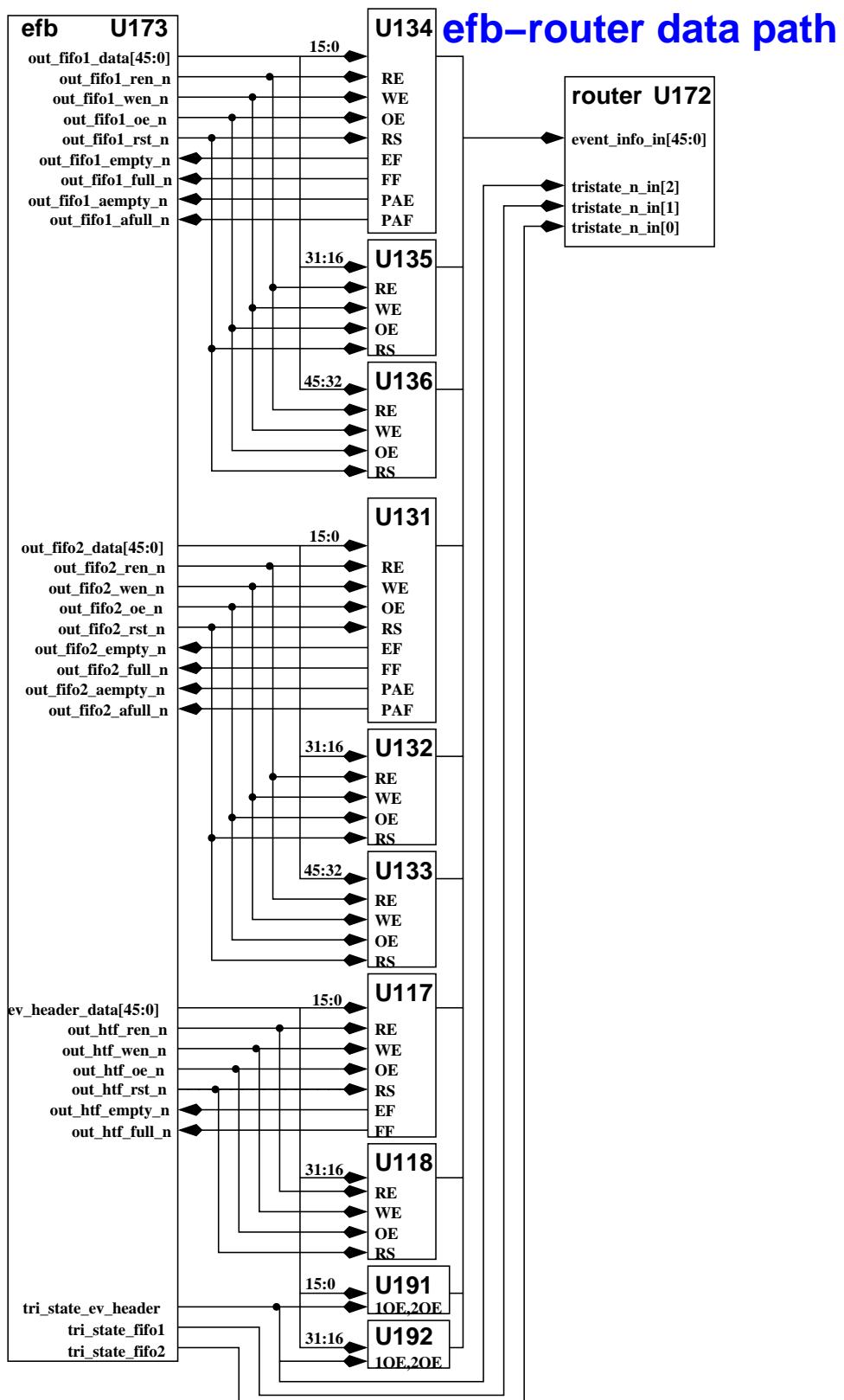
- **dedicated VHDL to send arbitrary bit patterns through the data path buffers and memories**

This requires more effort than the above test, but is still not too bad. VHDL to read and write the memories in the data path exists. That code is tied to the formatting and event building logic. It would have to be tied to logic controlling a simple pass-through register instead.

- A comprehensive set of test vectors which checks the data path, including control signals, will need to be provided. These test vectors will need to include error conditions. The test will need to generate flow control signals, header-only errors, buffer overflows, etc. This will likely be able to be performed using the actual VHDL, as is, by setting readout parameters appropriately.
- Without special hardware, it is difficult to check the backplane interface of the ROD prior to November's system test. All backplane signals are buffered, so it is possible to check at least the ROD outputs. There is a plan to build a loop-back board ROD production tests if needed.

formatter–efb data path





Work to do

- Continue work on VHDL simulation.
- Design and generate a comprehensive set of test vectors.
- Finish dedicated VHDL for testing.
- Write DSP test primitives (host and DSP code).
- Give more thought to the “mock fault challenge” proposed at the schematics review for testing the test system
- We need a logging system for the test results. Something simple for the initial set of boards, but by mid-2001 we should have a good production database.